PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 7 : G06F 9/355, 9/318, 12/02 — A1

(11) International Publication Number: WO 00/55723

(43) International Publication Date: 21 September 2000 (21.09.00)

(21) International Application Number: PCT/US00/05420

(22) International Filing Date: 29 February 2000 (29.02.00)

(30) Priority Data:
09/267,796 12 March 1999 (12.03.99) US

(71) Applicant *(for all designated States except US)*: INTEL CORPORATION [US/US]; 2200 Mission College Boulevard, Santa Clara, CA 95052 (US).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: SHAHIDZADEH, Shahrokh [US/US]; 16036 S.W. Waxwing Way, Beaverton, OR 97007 (US). BIGBEE, Bryant, E. [US/US]; 645 S.W. 195th Court, Aloha, OR 97006 (US). PAPWORTH, David, B. [US/US]; P.O. Box 6296, Beaverton, OR 97007 (US). BINNS, Frank [US/GB]; 1810 S.W. Manna Drive, Hillsboro, OR 97123 (US). COLWELL, Robert, P. [US/US]; 3594 N.W. Bronson Crest Loop, Portland, OR 97229 (US).

(74) Agents: MILLIKEN, Darren, J. et al.; Blakely, Sokoloff, Taylor & Zafman LLP, 7th floor, 12400 Wilshire Boulevard, Los Angeles, CA 90025 (US).

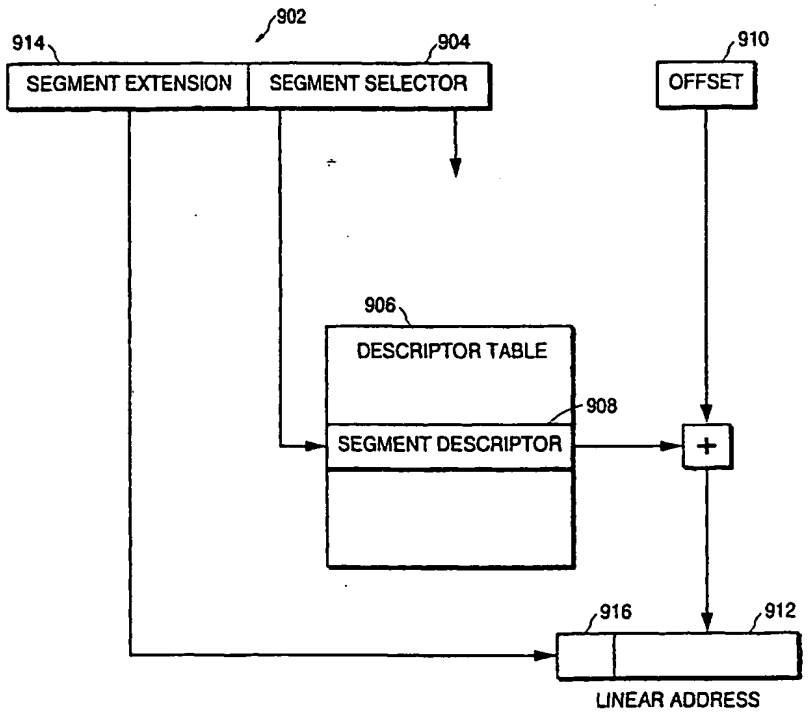
(81) Designated States: AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CR, CU, CZ, DE, DK, DM, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published
With international search report.

(54) Title: LINEAR ADDRESS EXTENSION AND MAPPING TO PHYSICAL MEMORY USING 4 AND 8 BYTE PAGE TABLE ENTRIES IN A 32-BIT MICROPROCESSOR

(57) Abstract

A microprocessor for providing an extended linear address of more than 32 bits. The extended linear address may be provided by concatenating a linear address with a segment selector extension, or by concatenating the values from two registers. Hierarchical translation of a linear address to a physical address is performed in which the number of levels in the hierarchy depends upon whether the linear address is an extended linear address.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece			TR	Turkey
BG	Bulgaria	HU	Hungary	ML	Mali	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MN	Mongolia	UA	Ukraine
BR	Brazil	IL	Israel	MR	Mauritania	UG	Uganda
BY	Belarus	IS	Iceland	MW	Malawi	US	United States of America
CA	Canada	IT	Italy	MX	Mexico	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NE	Niger	VN	Viet Nam
CG	Congo	KE	Kenya	NL	Netherlands	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NO	Norway	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	NZ	New Zealand		
CM	Cameroon			PL	Poland		
CN	China	KR	Republic of Korea	PT	Portugal		
CU	Cuba	KZ	Kazakstan	RO	Romania		
CZ	Czech Republic	LC	Saint Lucia	RU	Russian Federation		
DE	Germany	LI	Liechtenstein	SD	Sudan		
DK	Denmark	LK	Sri Lanka	SE	Sweden		
EE	Estonia	LR	Liberia	SG	Singapore		

Linear Address Extension and Mapping to Physical Memory Using 4 and 8 Byte Page Table Entries in a 32-Bit Microprocessor

Field

The present invention relates to microprocessor and computer systems, and more particularly, to virtual memory systems with extended linear address generation and translation.

Background

Most microprocessors make use of virtual or demand-paged memory schemes, where sections of a program's execution environment are mapped into physical memory as needed. Virtual memory schemes allow the use of physical memory much smaller in size than the linear address space of the microprocessor, and also provide a mechanism for memory protection so that multiple tasks (programs) sharing the same physical memory do not adversely interfere with each other.

Physical memory is part of a memory hierarchy system, which may be illustrated as part of a computer system shown in Fig. 1. Microprocessor **102** has a first level cache comprising instruction cache **104** and data cache **106.** Microprocessor **102** communicates with unified second level cache **108** via backside bus **110.** Second level cache **108** contains both instructions and data, and may physically reside on the chip die **102.** Caches **104** and **106** comprise the first level of the memory hierarchy, and cache **108** comprises the second level.

The third level of memory hierarchy for the exemplary computer system of Fig. 1 is indicated by memory **112.** Microprocessor **102** communicates with memory **112** via host processor (front side) bus **114** and chipset **116.** Chipset **116** may also provide graphics bus **118** for communication with graphics processor **120,** and serves as a bridge to other busses, such as peripheral component bus **122.** Secondary storage, such as disk unit **124,** provides yet another level in the memory hierarchy.

Fig. 2 illustrates some of the functional units within microprocessor **102,** including the instruction and data caches. In microprocessor **102,** fetch unit **202** fetches instructions from instruction cache **104,** and decode unit **206** decodes these instructions. For a CISC (Complex Instruction Set Computer) architecture, decode unit **206** decodes a complex instruction into one or more micro-instructions. Usually, these micro-instructions define a load-store type architecture, so that micro-instructions involving memory operations are simple load or store operations. However, the present invention

may be practiced for other architectures, such as for example RISC (Reduced Instruction Set Computer) or VLIW (Very Large Instruction Word) architectures.

For a RISC architecture, instructions are not decoded into micro-instructions. Because the present invention may be practiced for RISC architectures as well as CISC architectures, we shall not make a distinction between instructions and micro-instructions unless otherwise stated, and will simply refer to these as instructions.

Most instructions operate on several source operands and generate results. They name, either explicitly or through an indirection, the source and destination locations where values are read from or written to. A name may be either a logical (architectural) register or a location in memory. Renaming logical registers as physical registers may allow instructions to be executed out of order. In Fig. 2, register renaming is performed by renamer unit **208,** where RAT (Register Allocation Table) **210** stores current mappings between logical registers and physical registers. The physical registers are indicated by register file **212.**

Every logical register has a mapping to a physical register in physical register file **212,** where the mapping is stored in RAT **210** as an entry. An entry in RAT **210** is indexed by a logical register and contains a pointer to a physical register in physical register file **212.** Some registers in physical register file **212** may be dedicated for integers whereas others may be dedicated for floating point numbers, but for simplicity these distinctions are not indicated in Fig. 2.

During renaming of an instruction, the current RAT provides the required mapping for renaming the source logical register(s) of the instruction, and a new mapping is created for the destination logical register of the instruction. This new mapping evicts the old mapping in the RAT.

Renamed instructions are placed in instruction window buffer **216.** All instructions "in-flight" have an entry in instruction window buffer **216,** which operates as a circular buffer. Instruction window buffer **216** allows for memory disambiguation so that memory references are made correctly, and allows for instruction retirement in original program order. (For CISC architectures, a complex instruction is retired when all micro-instructions making up the complex instruction are retired together.)

For an instruction that writes its result to a memory location, data cache **106** (part of the memory hierarchy) is updated upon instruction retirement. For an instruction that writes its result to a logical register, no write need be done upon retirement because there are no registers dedicated as logical registers. (Physical register file **212** has the result of

the retiring instruction in that physical register which the destination logical register was mapped to when the instruction was renamed.)

Scheduler **218** schedules instructions to execution units **220** for execution. For simplicity, only memory execution unit **224** is explicitly indicated in execution units **220**. A load or store instruction is dispatched by scheduler **218** to AGU (Address Generation Unit) **222** for computation of a linear address, and memory execution unit **224** translates the linear address into a physical address and executes the load or store instruction. Memory execution unit may send data to or receive data from a forwarding buffer (not shown) rather than data cache **106,** where a forwarding buffer stores objects that may eventually be written to data cache **106** upon instruction retirement. The scheduling function performed by scheduler **218** may, for example, be realized by reservation stations (not shown) implementing Tomasulo's algorithm (or variations thereof) or by a scoreboard. Execution units **220** may retrieve data from or send data to register file **212,** depending upon the instruction to be executed.

In other embodiments of the present invention, the information content contained in the data structures of physical register field **212** and instruction window buffer **216** may be realized by different functional units. For example, a re-order buffer may replace instruction window buffer **216** and physical register file **212,** so that results are stored in the re-order buffer, and in addition, registers in a register file are dedicated as logical registers. For this type of embodiment, the result of an instruction that writes to a logical register is written to a logical register upon instruction retirement.

With most modern computer systems, a microprocessor refers to a memory location by generating a linear address, but an object is retrieved from a specific memory location by providing its physical address on an address bus, such as bus **114** in Fig. 1. Linear addresses may be the same as physical addresses, in which case address translation is not required. However, usually a virtual memory scheme is employed in which linear addresses are translated into physical addresses. In this case, a linear address may also be referred to as a virtual address. The linear address space is the set of all linear addresses generated by a microprocessor, whereas the physical address space is the set of all physical addresses.

For some microprocessor architectures, such as Intel® Architecture 32 bit (IA-32) microprocessors (Intel® is a registered trademark of Intel Corporation, Santa Clara, California), there is also another type of address translation in which a logical address is translated into a linear address. For these type of architectures, the instructions provide

logical address offsets, which are then translated to linear addresses by AGU **222** in Fig. 2. This extra stage of address translation may provide additional security, e.g., where application code cannot modify supervisory (operating system) code.

The mapping of a logical address to a linear address is illustrated in Fig. 3. A logical address comprises segment selector **302a** and offset **304.** Segment selector **302a** is stored in segment register **302,** which also contains descriptor cache **302b**. Segment selector **302a** points to segment descriptor **308** in descriptor table **306.** Descriptor table **306** provides a table of segment descriptors stored in memory. A segment descriptor provides a segment base address, so that a linear address is obtained by adding an offset to the base address provided by a segment descriptor, as indicated by summation **312.** In addition to providing a base address, a segment descriptor contains various other types of information, such as access rights and segment size. The base address, access rights, segment size, and other information, is cached in descriptor cache **302b**.

A virtual or demand-paged memory system may be illustrated as a mapping between a linear (virtual) address space and a physical address space, as shown in Fig. 4. In a virtual memory system, the linear and physical address spaces are divided into blocks of contiguous addresses, customarily referred to as pages if they are of constant size or are any of several fixed sizes. A typical page size may be 4KBytes, for example.

The mapping shown in Fig. 4 illustrates a generic two-level hierarchical mapping comprising directory tables and page tables. Page directory tables and page tables are stored in physical memory, and are usually themselves equal in size to a page. A page directory table entry (PDE) points to a page table in physical memory, and a page table entry (PTE) points to a page in physical memory. For the two-level hierarchical mapping of Fig. 4, a linear address comprises directory field **402,** table field **404,** and offset field **406.** A directory field is an offset to an PDE, a table field is an offset to an PTE, and an offset field is an offset to a memory location in a page.

In Fig. 4, page directory base register (PDBR) **408** points to the base address of page directory **410,** and the value stored in directory field **402** is added to the value stored in PDBR **408** to provide the physical address of PDE **412** in page directory **410.** PDE **412** in turn points to the base address of page table **414,** which is added to the value stored in table field **404** to point to PTE **416** in page table **414.** PTE **416** points to the base address of page **418,** and this page base address is added to the value stored in offset **406** to provide physical address **420.** Linear address **422** is thereby mapped to physical address **420.**

Accessing entries stored in page directories and page tables require memory bus transactions, which can be costly in terms of processor cycle time. However, because of the principle of locality, the number of memory bus transactions may be reduced by storing recent mappings between linear and physical addresses in a cache, called a translation look-aside buffer (TLB). There may be separate TLBs for instruction addresses and data addresses. Entries in a TLB are indexed by linear addresses. A hit in a TLB provides the physical address associated with a linear address. If there is a miss, then the memory hierarchy is accessed.(page walk) as indicated in Fig. 4 to obtain the translation of a linear address into a physical address.

Some IA-32 microprocessors employ several modes for translating linear addresses into physical addresses, and we shall consider three such modes herein referred to as modes A, B, and C. Mode A supports a 32 bit physical address space with 4KB page sizes. Mode B supports a 32 bit physical address space with either 4KB or 4MB page sizes. For modes A and B, the page and directory table entries are each 4 bytes. Mode C supports a 36 bit physical address space for a physical address size of 64GB (physical address extension) with either 4KB or 2MB page sizes. For mode C, the page and directory table entries are each 8 bytes. For each mode, the page and directory tables are equal in size to a page. All modes are for translating 32 bit linear addresses.

Mode A is illustrated in Fig. 5, where the first 12 bits of a linear address are used as an offset to a physical address within a page frame, the next 10 bits of the linear address are used as an offset into a page table, and the highest 10 bits of the linear address are used as an offset into a page directory. For example, in Fig. 5, PTE **502** in page table **504** pointed to by table field **506** of the linear address provides the address of the desired page frame in physical memory, and when concatenated with offset **508** of the linear address provides the physical address of the desired object. The PDBR register, page directory entries, and page table entries each provide the upper 20 bits of a 32 bit address, so that page directories, page tables, and pages are each forced to be aligned on 4KB boundaries.

Mode B for 4MB page sizes is illustrated in Fig. 6. (For 4KB page sizes, mode B is similar to mode A. The first 22 bits of the linear address provides the offset into a physical 4MB page frame, and the highest 10 bits of the linear address provides the offset into a page table. Note that mode B with 4MB page sizes requires only one level of address translation. A PDE in the page directory of Fig. 6 provides the upper 10 bits of a 32 bit address to force pages to be aligned on 4MB boundaries.

Mode C for 4KB page sizes is illustrated in Fig. 7. This involves a third level of address translation provided by page directory pointer table (PDPT) **702.** Each entry in PDPT **702** is 8 bytes, and there are 4 entries in a PDPT. PDBR **704** provides the upper 27 bits of a 32 bit address pointing to the base of a PDPT so that PDPTs are forced to be aligned on 32 byte boundaries. Each entry in the PDPT, page directory, and page table provides the upper 24 bits of a 36 bit address so that page directories, page tables, and pages are forced to be aligned on 4KB boundaries.

Mode C for 2MB page sizes is illustrated in Fig. 8. Only two levels of address translation are required, where again a four entry PDPT is used to point to a page directory. Entries in the page directory provide the upper 15 bits of a 36 bit address so that pages are forced to be aligned on 2MB boundaries.

The page structure described in Figs. 7 and 8 for Mode C allows up to 4GB of the 64GB extended address space to be addressed at one time. To address other 4GB sections of the extended address space, a different entry may be placed in the PDBR register so as to point to a different PDPT, or entries in the PDPT may be changed. Further details of address translation for the IA-32 architecture may be found in the Intel Architecture Developer's Manual for the Pentium® Pro, Vol. 3, available from Intel Corporation. (Pentium® Pro is a registered trademark of Intel Corporation.)

Increasing the linear address space of a microprocessor provides larger user and system space and reduces the burden associated with linear address exhaustion for a larger physical address space. Increasing the word size of a microprocessor, e.g., from 32 bits to 64 bits, to provide a larger linear address space is a major engineering design task. It may therefore be of economic utility to increase the linear address space of an existing microprocessor design without increasing its word size. Furthermore, it may be advantageous for a microprocessor with increased linear address space to be backward compatible with code designed for the original sized linear address space and supported paging structures.

Summary

Embodiments of the present invention are directed to computers and microprocessors providing an extended linear address space. In one embodiment, an extended linear address is generated in which its lower portion is based upon an offset and a segment selector and its upper portion is based upon a segment extension. In another embodiment, a linear address is generated by concatenating values stored in two registers. Other embodiments provide for translating a linear address to a physical

address by accessing page directories such that the level of hierarchical translation depends upon whether there is linear address extension.

Brief Description of the Drawings

Fig. 1 provides a prior art diagram of a computer system.

Fig. 2 provides a prior art diagram of a microprocessor.

Fig. 3 provides a prior art illustration for translating a logical address to a linear address.

Fig. 4 provides a prior art illustration for translating a linear address to a physical address.

Fig. 5 provides a prior art illustration for translating a 32 bit linear address to a 32 bit physical address with 4-KByte paging.

Fig. 6 provides a prior art illustration for translating a 32 bit linear address to a 32 bit physical address with 4-MByte paging.

Fig. 7 provides a prior art illustration for translating a 32 bit linear address to a 36 bit physical address with 4-KByte paging.

Fig. 8 provides a prior art illustration for translating a 32 bit linear address to a 36 bit physical address with 2-MByte paging.

Fig. 9 provides an exemplary embodiment for providing an extended linear address.

Fig. 10 provides another exemplary embodiment for providing an extended linear address.

Fig. 10a provides an exemplary implementation of Fig. 10.

Fig. 11 provides an exemplary embodiment for translating a 42 bit linear address to a 36 bit physical address with 4-KByte paging and 4 byte entries.

Fig. 12 provides an exemplary embodiment for translating a 42 bit linear address to a 36 bit physical address with 4-MByte paging and 4 byte entries.

Fig. 13 provides an exemplary embodiment for translating a 42 bit linear address to a 36 bit physical address with 4-KByte paging and 8 byte entries.

Fig. 14 provides an exemplary embodiment for translating a 42 bit linear address to a 36 bit physical address with 2-MByte paging and 8 byte entries.

Detailed Description of Embodiments

Embodiments of the present invention provide for a backward compatible, extended linear address space by utilizing one or more opcodes to indicate when extended linear addressing is to be utilized. In one embodiment, an opcode indicates

whether an LAE (Linear Address Extension) bit in a microprocessor register is to be set. If the LAE bit is set, then AGU **222** translates logical addresses to extended linear addresses.

Fig. 9 illustrates an embodiment for translating logical addresses to extended linear addresses. Segment register **902** is extended beyond that which is required to select a segment descriptor, as seen in Fig. 9. A portion of segment register **902,** denoted by segment selector **904,** is used to select descriptor table **906** and segment descriptor **908** so as to provide a base address as discussed previously, and an offset value in offset register **910** is added to the base address to provide a lower portion of the extended linear address, indicated by **912.** A portion of segment register **902** not used to select segment descriptor **908,** denoted by segment extension **914,** forms the upper portion of the extended linear address, denoted by **916.** When **914** is added or concatenated with lower portion linear address **912,** the extended linear address is obtained.

In another embodiment, instructions provide the extended linear address via their source registers, where the extended linear address is obtained by concatenating the values stored in the source registers. For example, a new instructions for loading, storing, adding, and exchanging objects in memory may be introduced in the instruction set. These new instructions are decoded by decoder **206** into one or more microinstructions, where a microinstruction specifies an extended linear address via source registers in its opcode.

This procedure is illustrated in the flow diagram of Fig. 10. In step **1002**, the values in the source register named by the decoded instruction are concatenated to form the extended linear address, provided the decoded instruction is an instruction that belongs to the set of instructions operating in the extended linear address space. In step **1004**, when a decoded instruction is to operate in the original linear address space, then the offset provided by the instruction is added to the base address provided by the segment descriptor to obtain the linear address.

Fig. 10a provides an implementation of Fig. 10. Multiplexers **1006**, **1008**, and **1010** select their inputs based upon whether there is an extended linear address microinstruction (LAEuop line is asserted). If the LAEuop line is not asserted, multiplexers **1006** and **1008** provide to AGU **222** an instruction queue immediate and segment base address, respectively, so that the linear address may be computed in a conventional manner. If, however, LAEuop is asserted, then multiplexer **1010** provides

the concatenation of the contents of registers *R*1 and *R*2 so that an extended linear address is obtained.

Once an extended linear address is generated, it is translated to a physical linear address. Embodiments of the present invention provide for this address translation by introducing an extra level of translation into the address translation hierarchy, where this extra level of translation is conditionally utilized provided extended linear addressing is indicated.

Fig. 11 illustrates an embodiment for extended linear address translation with 4KB paging. In the specific example of Fig. 11, an extended linear address is 42 bits, where the highest 10 bits are used as an offset into page directory **1102**. The base address for page directory **1102** is provided by PDBR **1104**. Mux (multiplexer) **1106** is used symbolically in Fig. 11 to indicate that if extended linear addressing is indicated, then the PDE provided by page directory **1102** is used as the base address for the next lower level of address translation, which is page directory **1108**. Extended linear addressing may be indicated, for example, if an LAE bit is set or if not all of the upper 10 bits of the linear address are zero. If extended linear addressing is not supported, then mux **1106** symbolically indicates that PDBR **1104** is used to point to the base address of page directory **1108**. Directory and page table entries are each 4 bytes. PTE **1110** provides the upper 24 bits of a 36 physical address, which when concatenated with 12 bits from offset **1112** provides a 36 bit address. The physical address space is 64-GBytes.

An embodiment for extended linear address translation with 4MB paging is shown in Fig. 12. Page directory **1202** provides an extra level of address translation, conditional upon whether extended linear addressing is indicated. Because of the page size, only two levels of page directories are utilized for translating an extended linear address to a physical address. Each page directory entry in Fig. 12 is 4 bytes, and the physical address space is 64-GBytes.

An embodiment for extended linear address translation with 4KB paging in an extended physical address space of 64-GBytes is shown in Fig. 13. As in mode C in Fig. 7, when extended linear address translation is not indicated, PDBR **1302** is used to point to the base address of PDPT **1304**, which is usually kept in a cache. To support extended linear address translation, PDBR **1302** is used to point to the base address of page directory **1306**, and bit positions 30 through 38 (Addr[38:30]) of the extended linear address provide the offset into page directory **1306**. Note that the first four entries in page directory **1306** are also cached in PDPT **1304**. For Fig. 13, directory and page table

entries are each 8 bytes, and the number of entries in each directory and page is $2^9 = 512$ so that each directory and page is 4KB in size. Because only 9 bits are used as an offset into page directory **1306**, bits above position 38 in the linear address are not used in this embodiment. Consequently, if the linear address register is 42 bits, extended address translation is provided for 42 bit linear addresses with the highest three bits equal to zero.

An embodiment for extended linear address translation with 2MB paging in an extended physical address space of 64-GBytes is shown in Fig. 14, and should be self-explanatory. Entries in the page directories of Fig. 14 are 8 bytes, so that as in Fig. 13 the linear address bits above position 38 are zero.

Various modifications may be made to the disclosed embodiments without departing from the scope of the invention as claimed below.

What is claimed is:

1. A microprocessor comprising:
a decoder to decode instructions so as to provide an offset;
a segment register to store a segment selector and a segment extension; and
an address generation unit to generate an extended linear address, wherein the extended linear address comprises a lower portion and an upper portion, wherein the lower portion is based upon the offset and the segment selector, and the upper portion is based upon the segment extension.

2. The microprocessor as set forth in claim 1, wherein the address generation unit provides the lower portion as the sum of the offset and a base address, wherein the base address is provided by a segment descriptor from a descriptor table stored in memory and pointed to by the segment selector, wherein the upper portion is equal to the segment extension.

3. The microprocessor as set forth in claim 2, wherein the microprocessor is a 32 bit processor and the extended linear address has more than 32 bits.

4. A computer comprising:
a system bus;
memory coupled to the system bus; and
a microprocessor coupled to the system bus, the microprocessor comprising
a decoder to decode instructions so as to provide an offset;
a segment register to store a segment selector and a segment extension; and
an address generation unit to generate an extended linear address, wherein the extended linear address comprises a lower portion and an upper portion, wherein the lower portion is based upon the offset and the segment selector, and the upper portion is based upon the segment extension.

5. The computer as set forth in claim 4, wherein the address generation unit provides the lower portion as the sum of the offset and a base address, wherein the base address is provided by a segment descriptor from a descriptor table stored in the memory and pointed to by the segment selector, wherein the upper portion is equal to the segment extension.

6. The computer as set forth in claim 5, wherein the microprocessor is a 32 bit processor and the extended linear address has more than 32 bits.

7. A microprocessor comprising:

a register file;

a decoder to decode instructions belonging to an instruction set, wherein the instruction set includes an instruction to specify an extended linear address, wherein the instruction names a first source register in the register file and a second source register in the register file; and

an address generation unit to generate the extended linear address as a concatenation of values stored in the first and second source registers.

8. The microprocessor as set forth in claim 7, wherein the microprocessor is a 32 bit processor and the extended linear address has more than 32 bits.

9. A computer comprising:

a system bus;

memory coupled to the system bus; and

a microprocessor coupled to the system bus, the microprocessor comprising

a register file;

a decoder to decode instructions belonging to an instruction set, wherein the instruction set includes an instruction to specify an extended linear address, wherein the instruction names a first source register in the register file and a second source register in the register file; and

an address generation unit to generate the extended linear address as a concatenation of values stored in the first and second source registers.

10. The computer as set forth in claim 9, wherein the microprocessor is a 32 bit processor and the extended linear address has more than 32 bits.

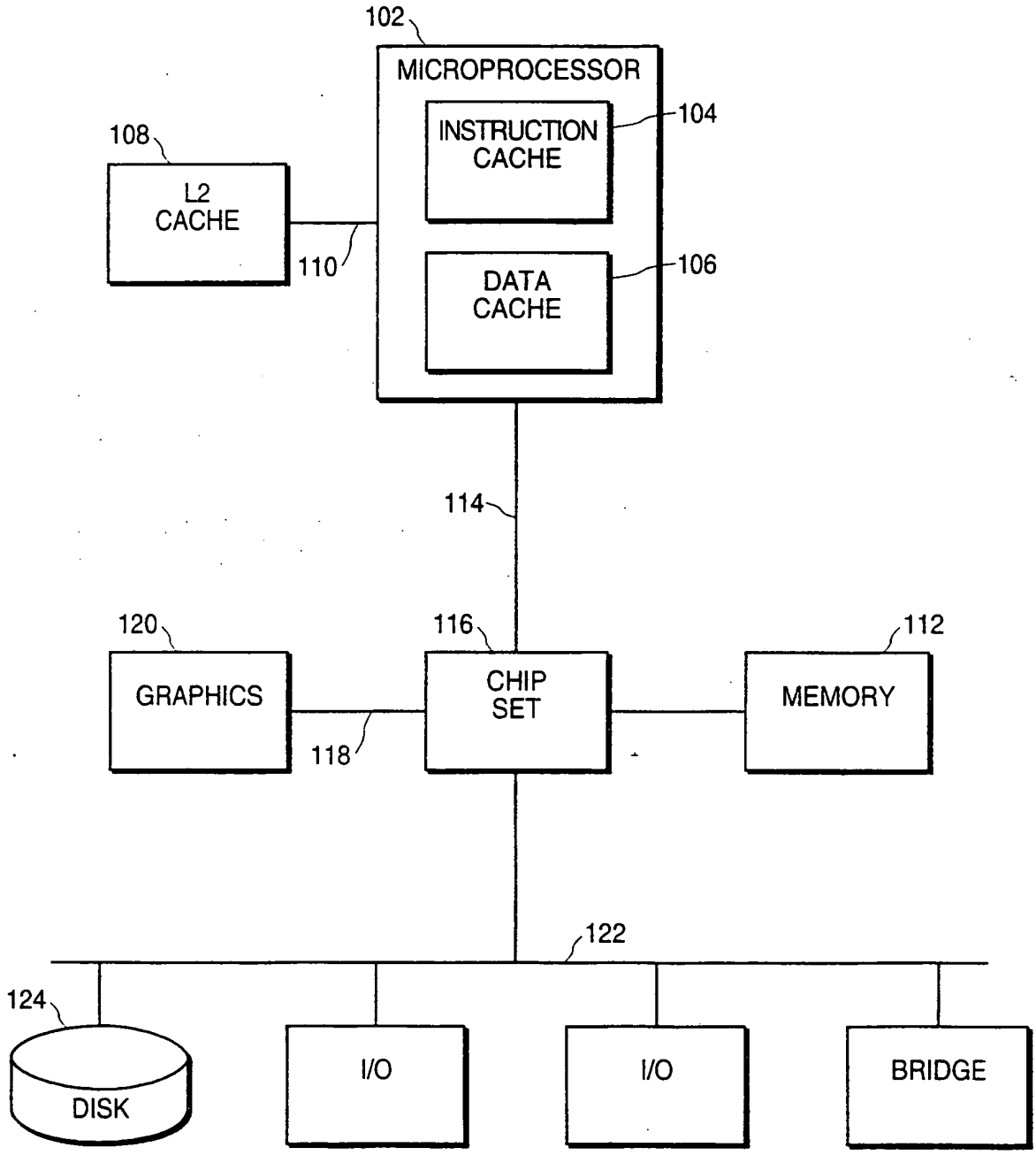


FIG.1 (PRIOR ART)

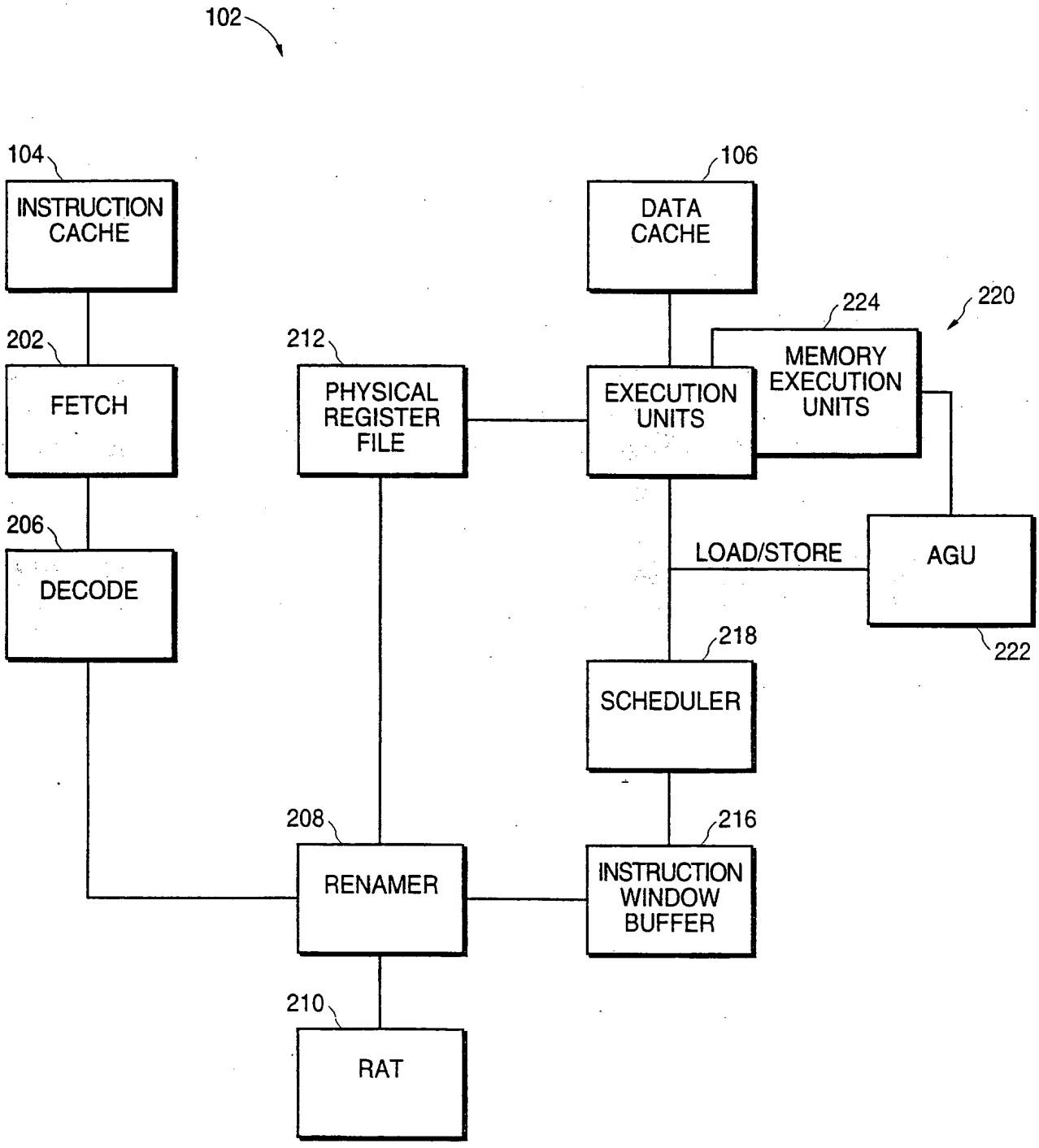


FIG.2 (PRIOR ART)

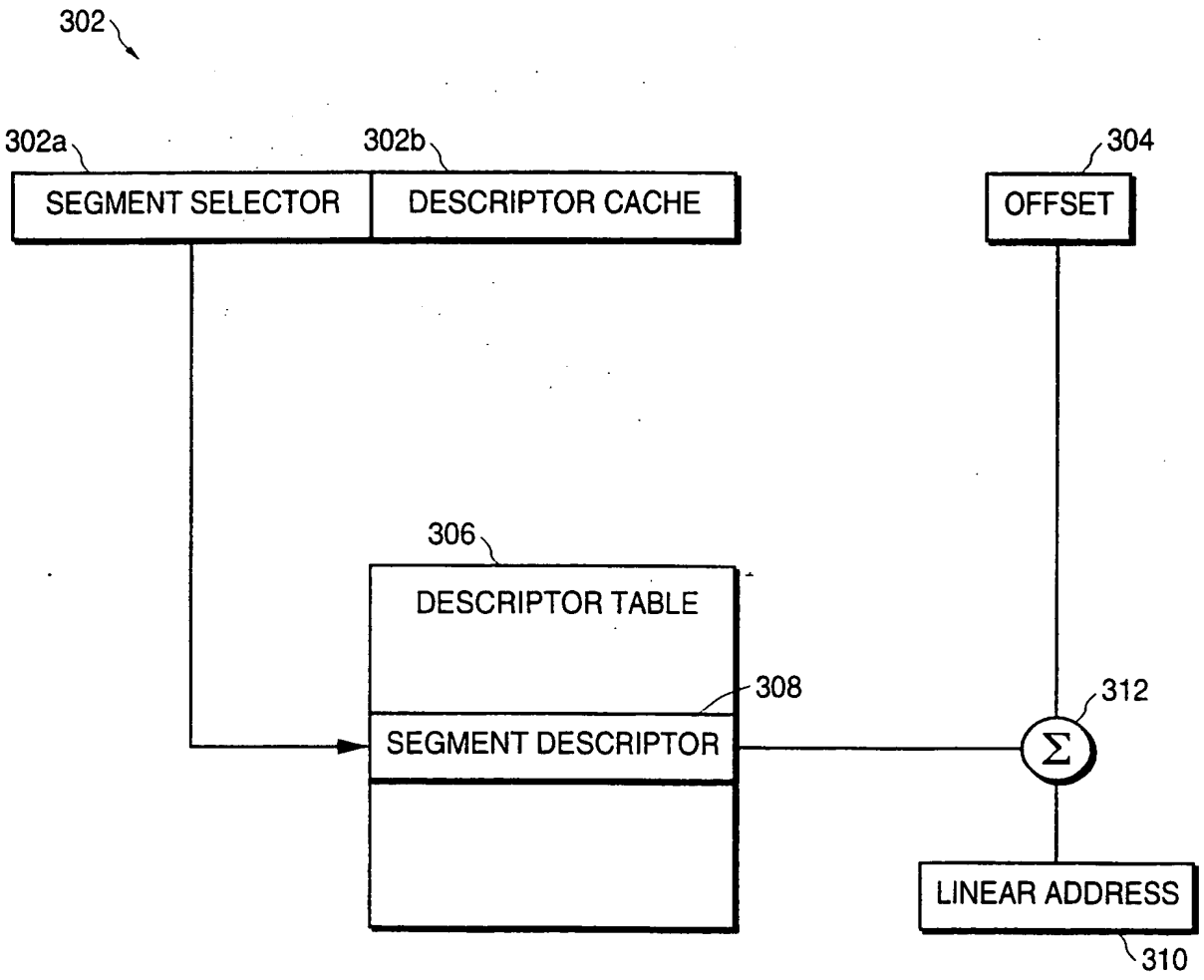


FIG.3 (PRIOR ART)

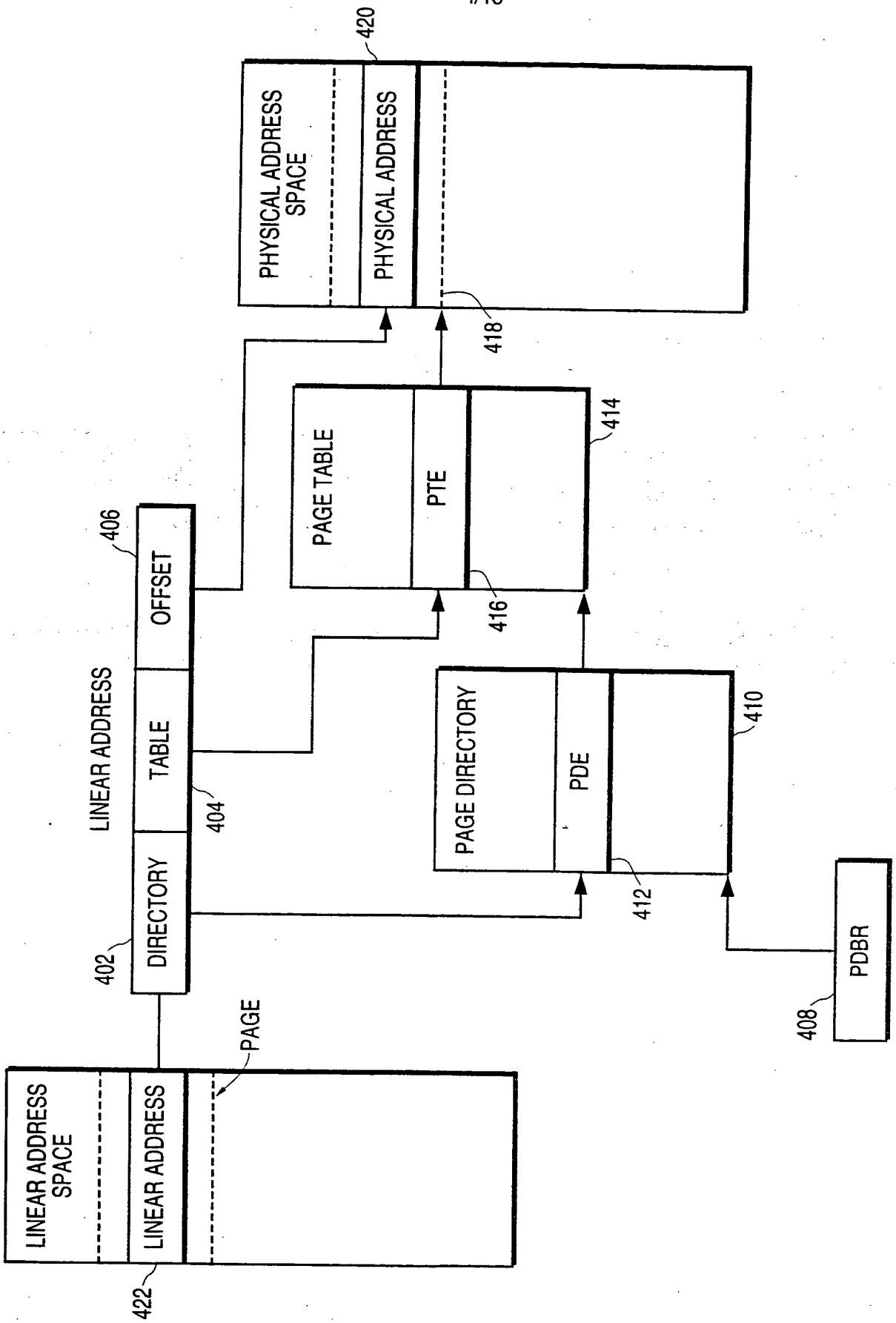


FIG.4 (PRIOR ART)

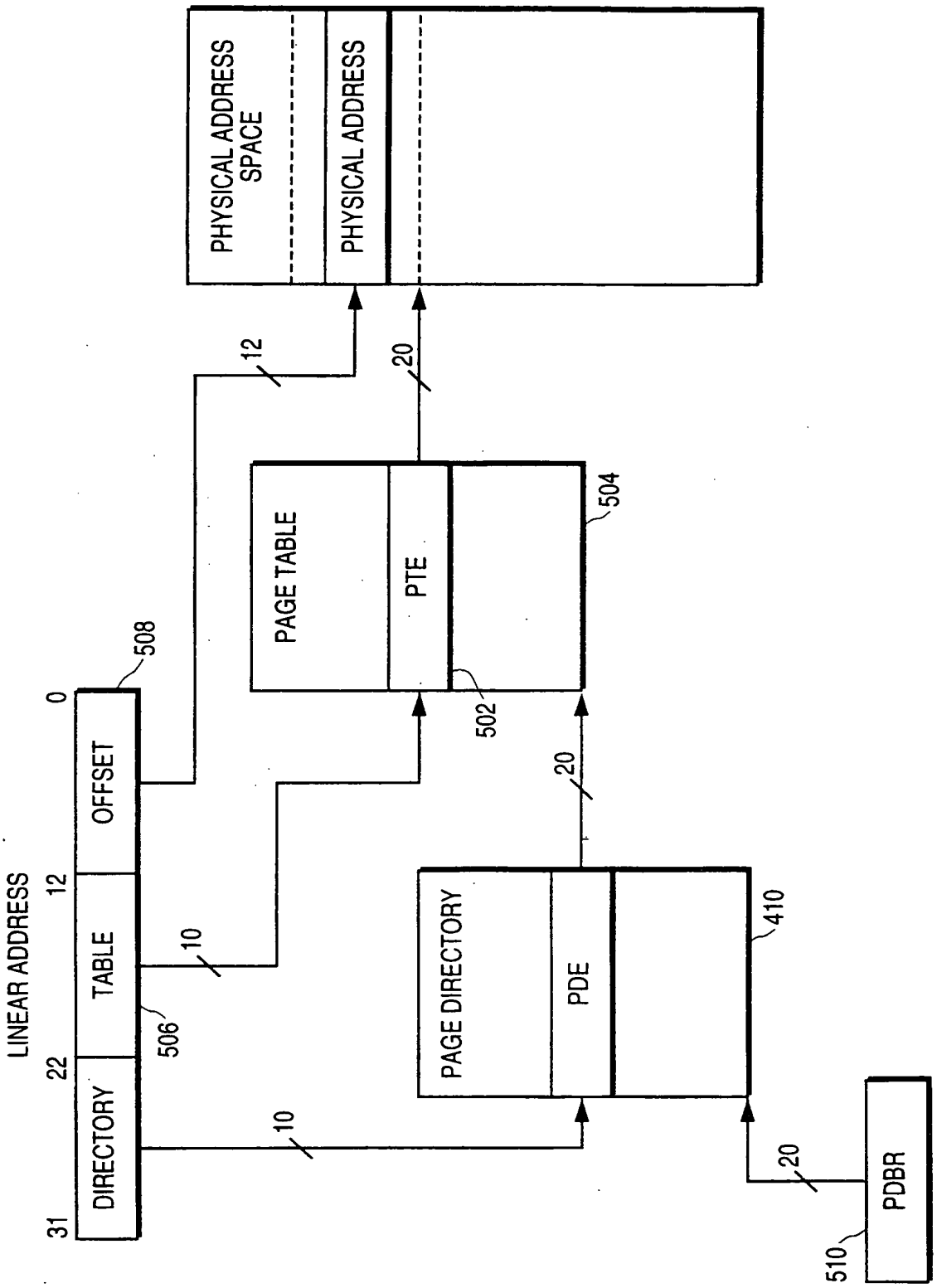


FIG.5 (PRIOR ART)

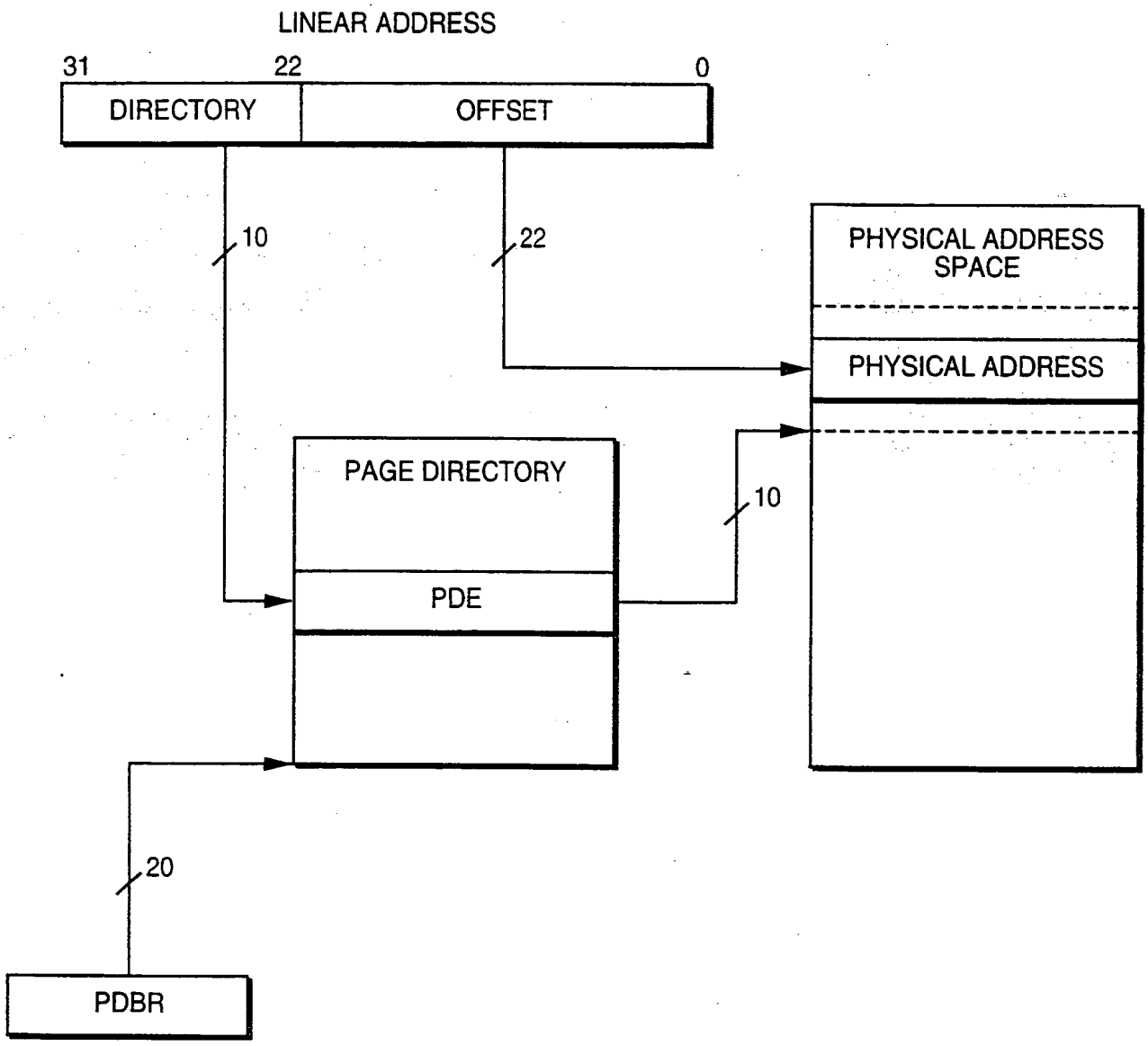


FIG.6 (PRIOR ART)

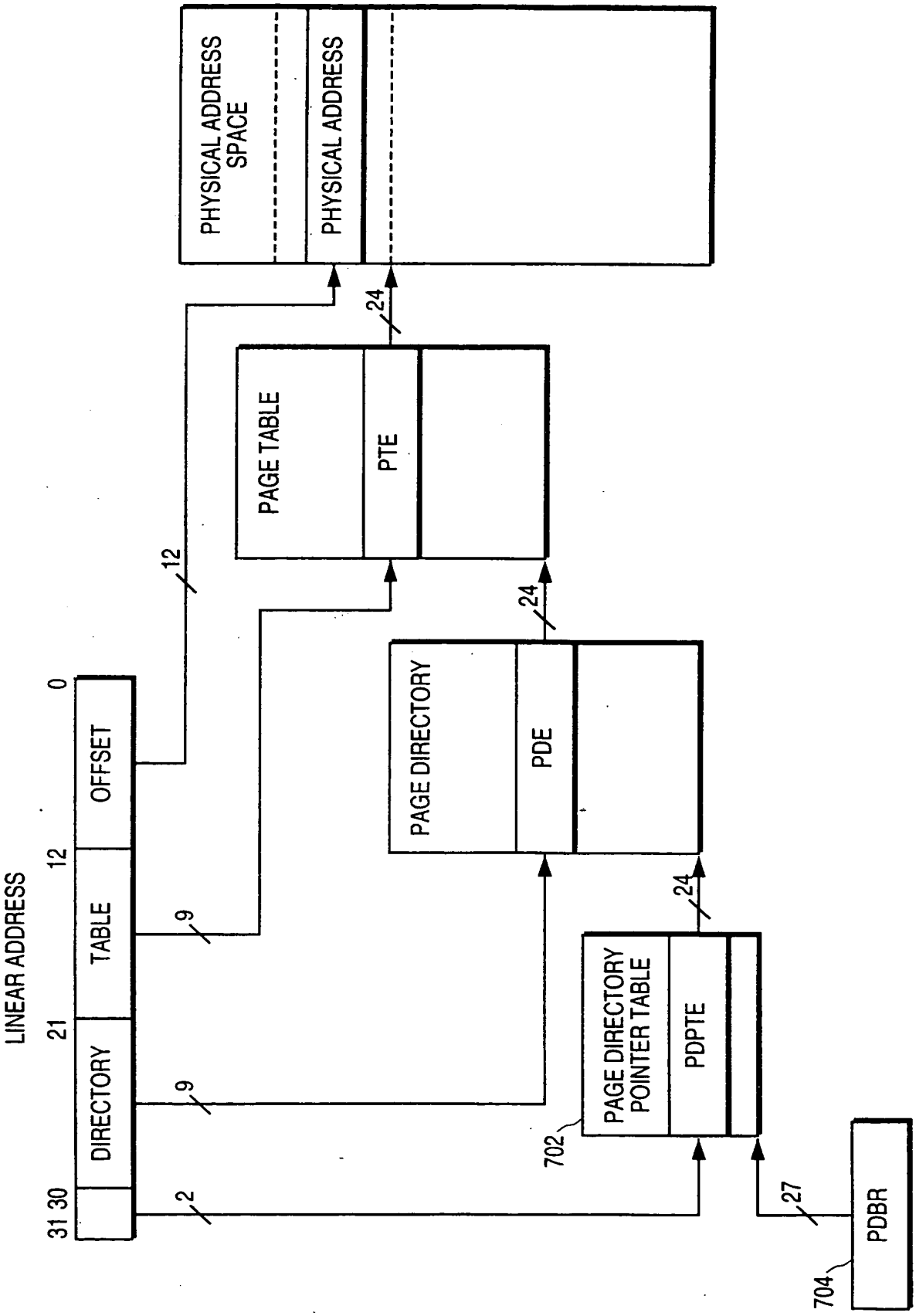


FIG.7 (PRIOR ART)

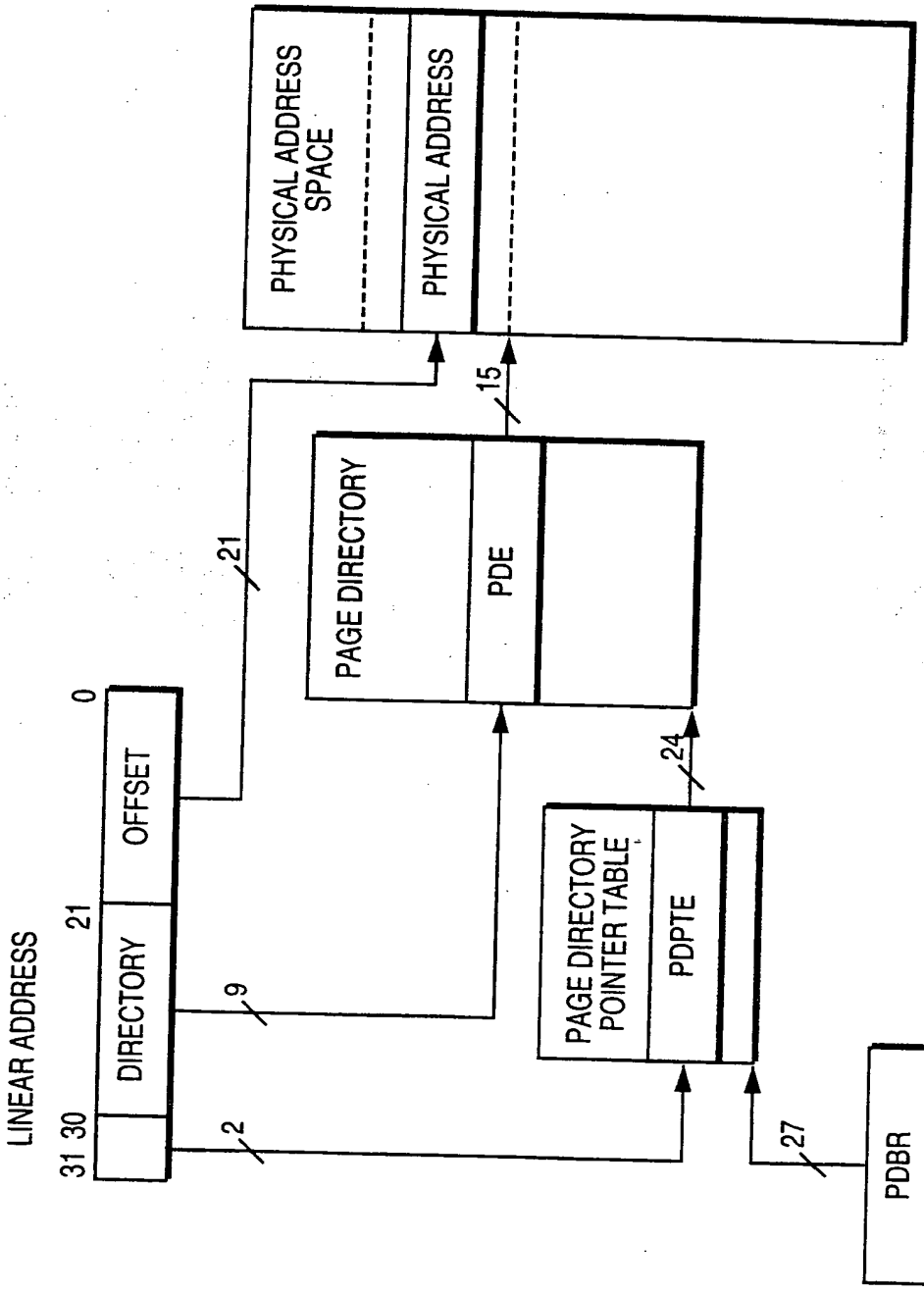


FIG.8 (PRIOR ART)

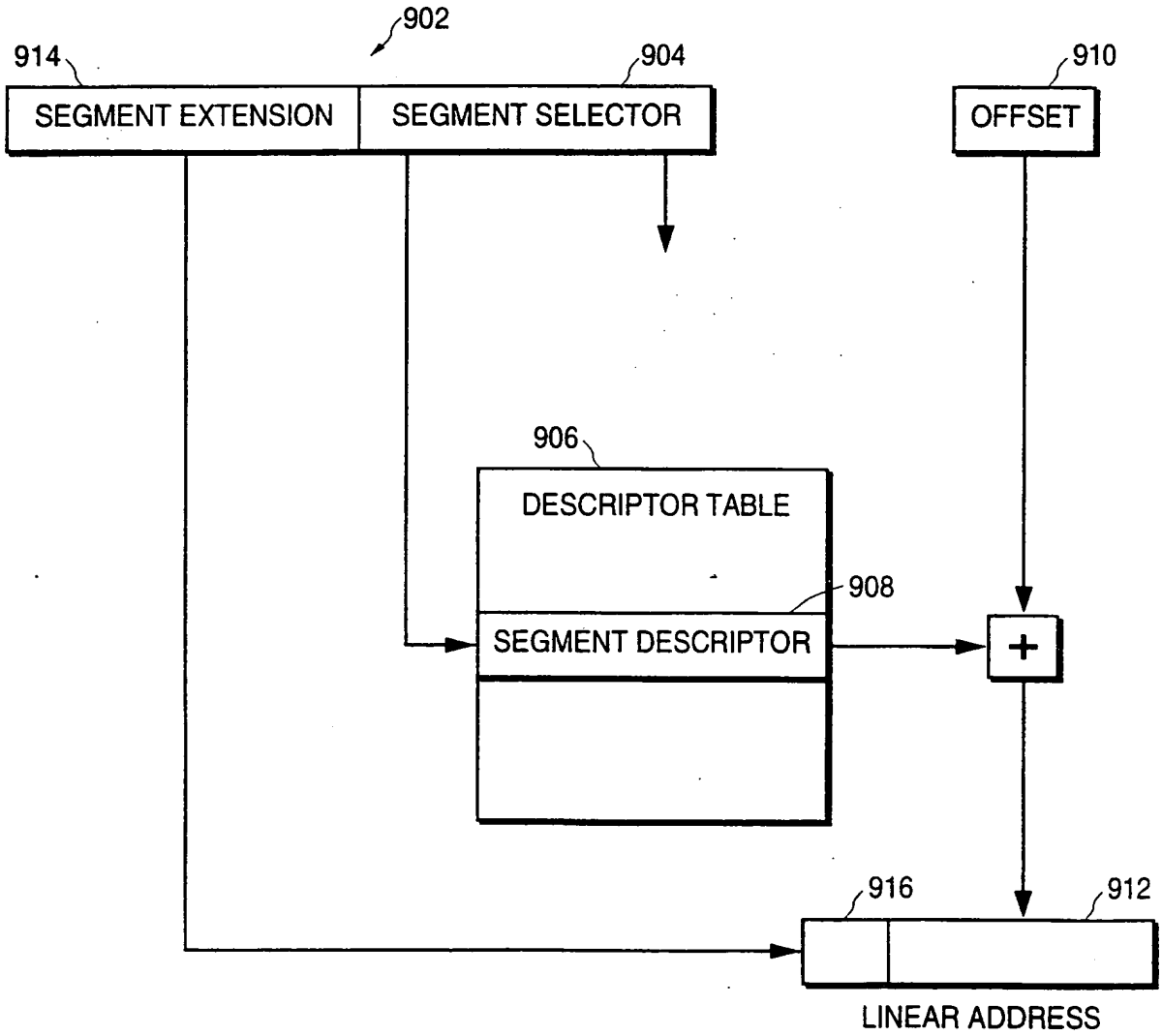

902
914
904
910
SEGMENT EXTENSION
SEGMENT SELECTOR
OFFSET
906
DESCRIPTOR TABLE
908
SEGMENT DESCRIPTOR
+
916
912
LINEAR ADDRESS

FIG.9

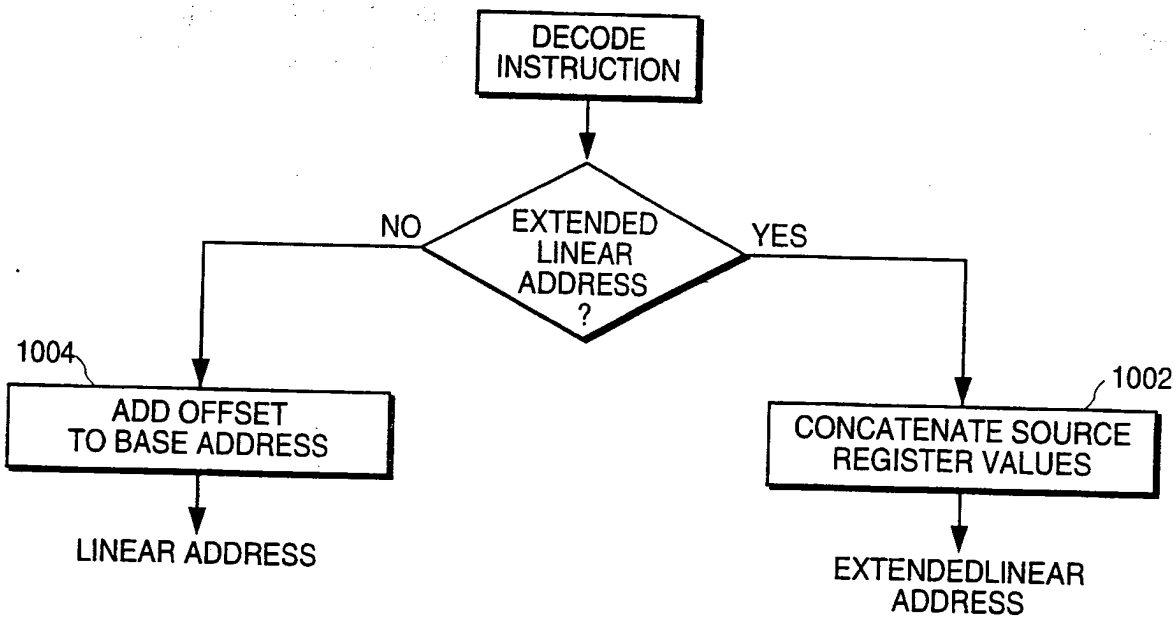

DECODE INSTRUCTION
EXTENDED LINEAR ADDRESS ?
NO
YES
1004
ADD OFFSET TO BASE ADDRESS
LINEAR ADDRESS
1002
CONCATENATE SOURCE REGISTER VALUES
EXTENDEDLINEAR ADDRESS

FIG.10

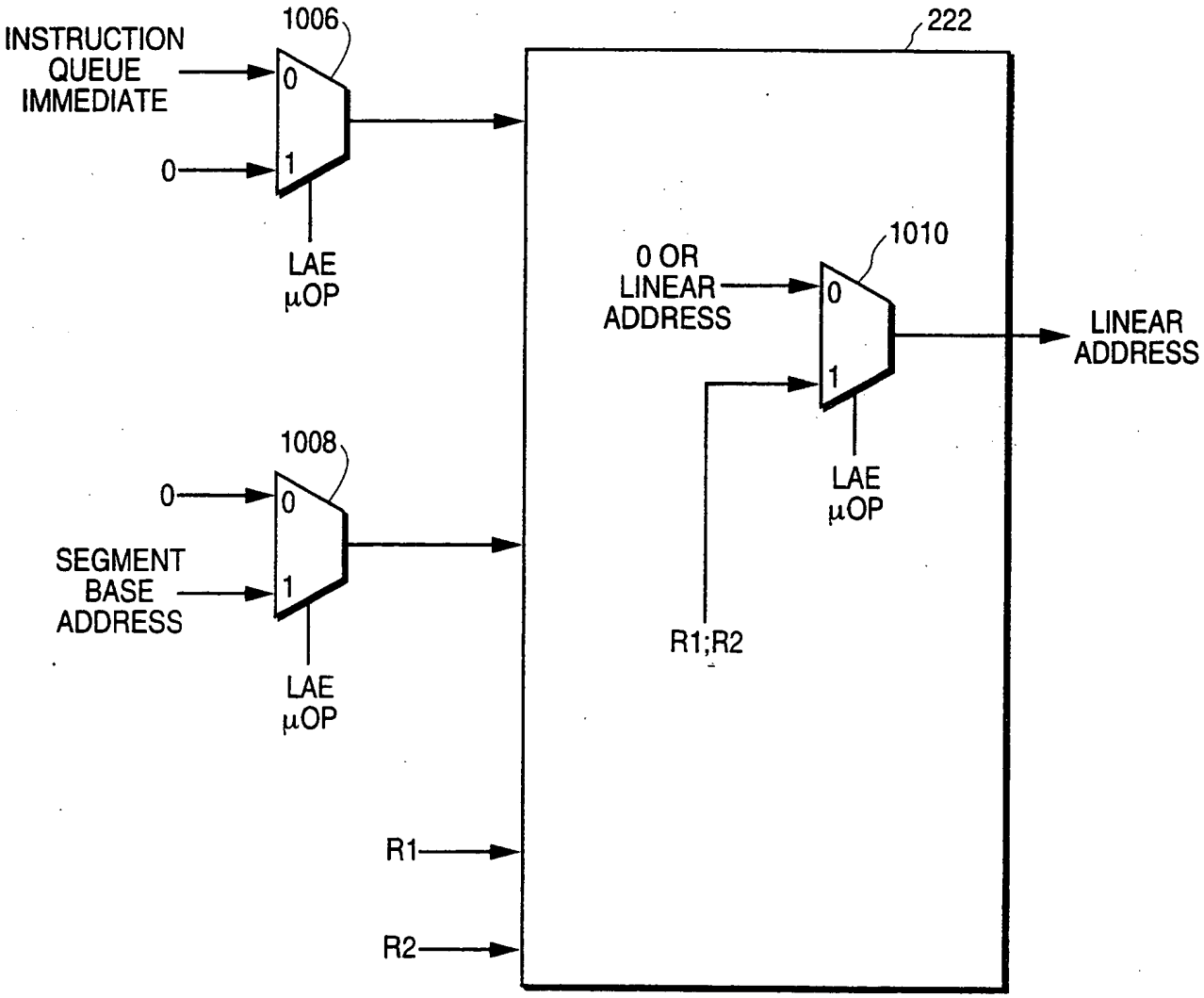

INSTRUCTION QUEUE IMMEDIATE
0
1006
0
1
LAE μOP
222
0 OR LINEAR ADDRESS
1010
0
1
LINEAR ADDRESS
LAE μOP
R1;R2
0
1008
0
1
SEGMENT BASE ADDRESS
LAE μOP
R1
R2

FIG 10A

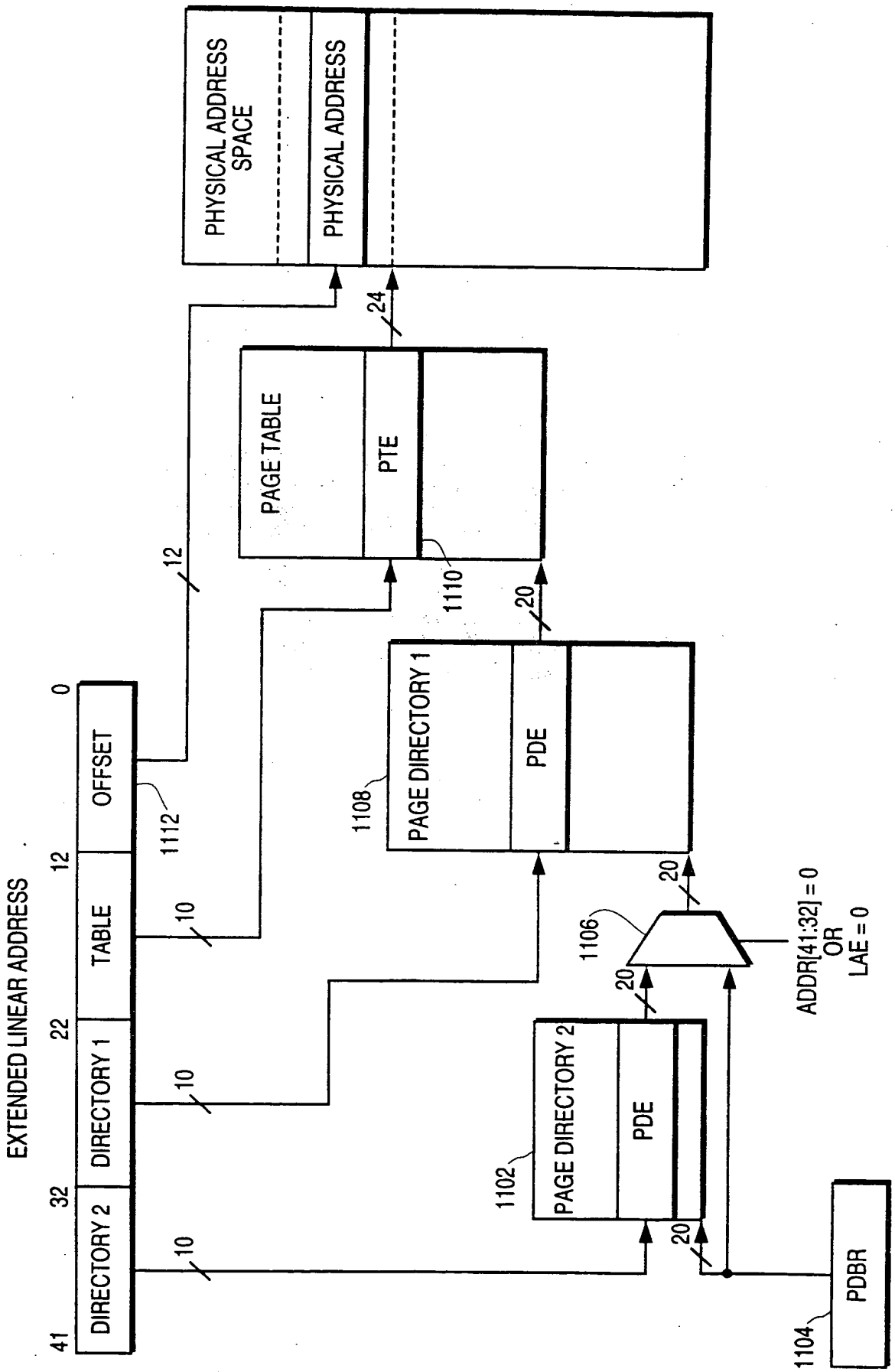

EXTENDED LINEAR ADDRESS
41
32
22
12
0
DIRECTORY 2
DIRECTORY 1
TABLE
OFFSET
10
10
10
12
1112
PDBR
1104
PAGE DIRECTORY 2
PDE
1102
PAGE DIRECTORY 1
PDE
1108
PAGE TABLE
PTE
1110
20
20
20
20
1106
ADDR[41:32] = 0
OR
LAE = 0
24
PHYSICAL ADDRESS
PHYSICAL ADDRESS SPACE

FIG. 11

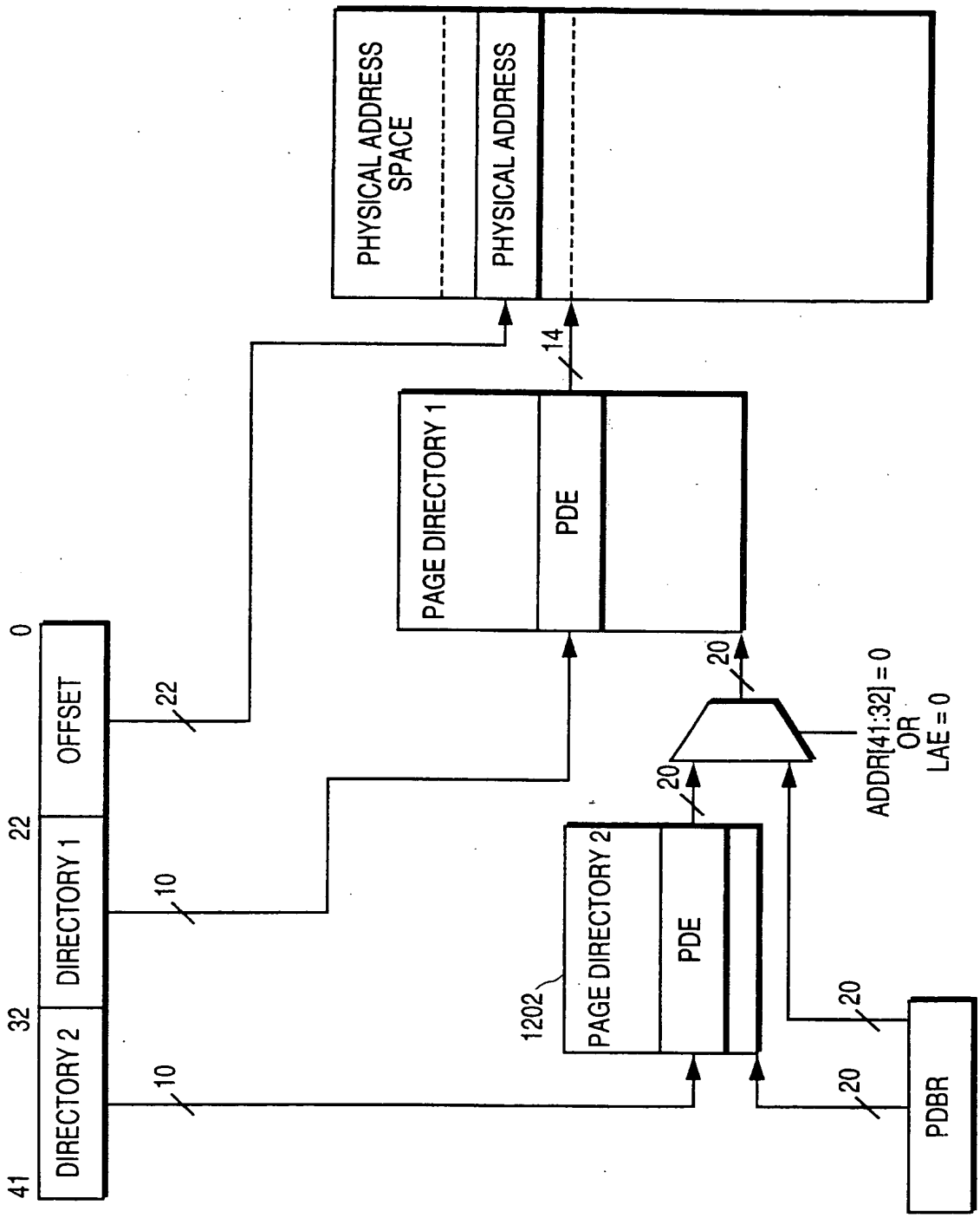


FIG.12

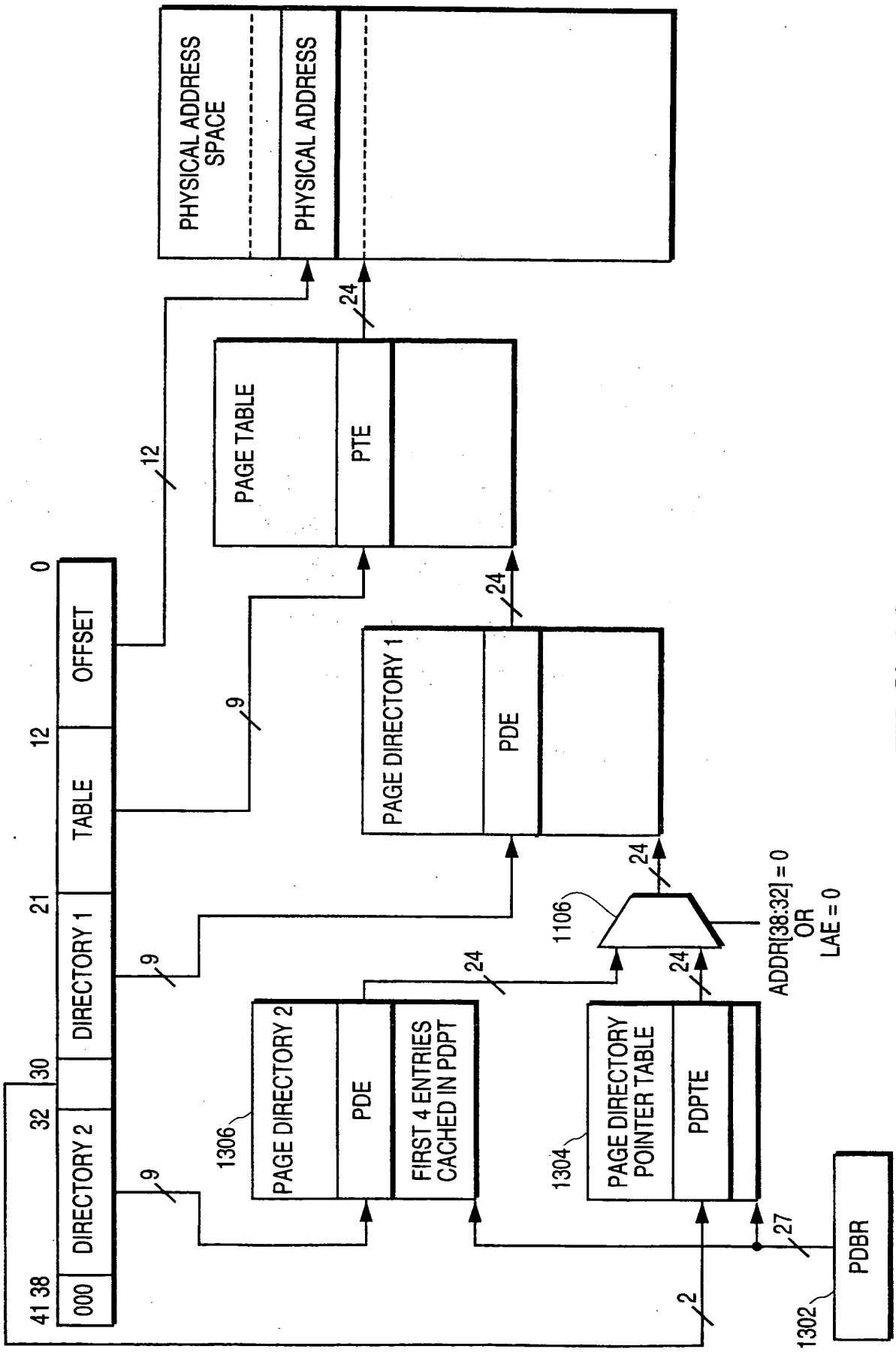


FIG.13

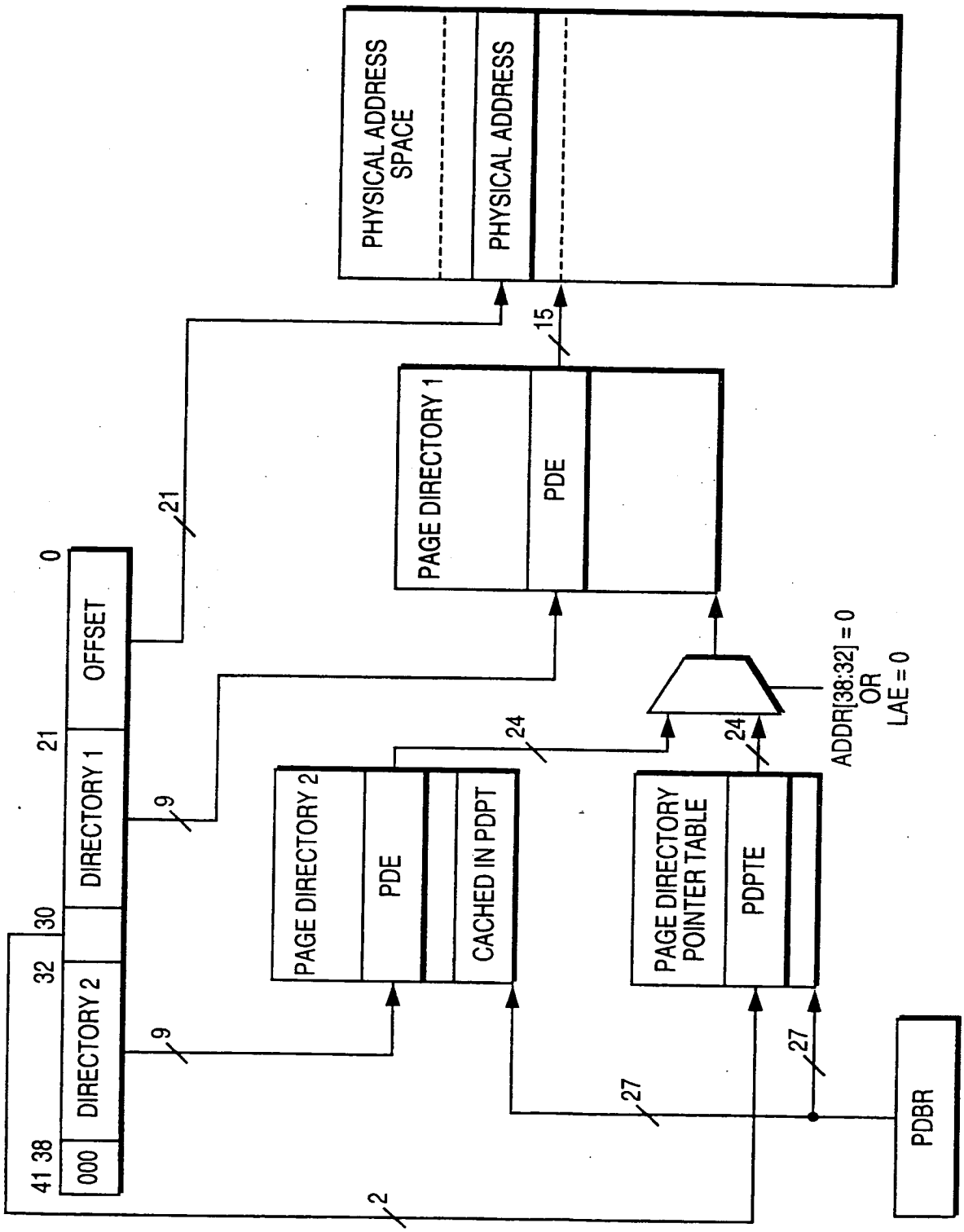

PHYSICAL ADDRESS SPACE
PHYSICAL ADDRESS
PAGE DIRECTORY 1
PDE
15
21
0
OFFSET
21
DIRECTORY 1
30
32
DIRECTORY 2
41 38
000
9
9
2
PAGE DIRECTORY 2
PDE
CACHED IN PDPT
24
24
PAGE DIRECTORY POINTER TABLE
PDPTE
ADDR[38:32] = 0
OR
LAE = 0
27
27
PDBR

FIG.14

INTERNATIONAL SEARCH REPORT

Inte onal Application No
PCT/US 00/05420

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 G06F9/355 G06F9/318 G06F12/02

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal, PAJ

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	EP 0 530 682 A (IBM) 10 March 1993 (1993-03-10) column 1, line 56 -column 6, line 6 ---	1-6
X	US 5 566 308 A (BENDELAC CHAIM ET AL) 15 October 1996 (1996-10-15) column 2, line 38 -column 47; figure 6 --- -/--	7,9

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
28 June 2000	05/07/2000

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31–70) 340–2040, Tx. 31 651 epo nl, Fax: (+31–70) 340–3016	Nielsen, O

1

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Int. ional Application No
PCT/US 00/05420

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
A	C RAY PENG ET AL: "THE POWER ARCHITECTURETM: 64-BIT POWER WITH 32-BIT COMPATIBILITY" DIGEST OF PAPERS OF THE COMPUTER SOCIETY COMPUTER CONFERENCE (SPRING) COMPCON,US,LOS ALAMITOS, IEEE COMP. SOC. PRESS, vol. CONF. 40, 1995, pages 300-307, XP000545443 ISBN: 0-7803-2657-1 page 300, right-hand column, line 15 -page 303, left-hand column, line 5 ---	1-10
A	US 4 679 140 A (KAGIMASA TOYOHIKO ET AL) 7 July 1987 (1987-07-07) column 2, line 50 - line 68 -----	1-10

Form PCT/ISA/210 (continuation of second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

information on patent family members

Inte .onal Application No
PCT/US 00/05420

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0530682 A	10-03-1993	US 5423013 A	06-06-1995
		AT 180338 T	15-06-1999
		CA 2075305 A,C	05-03-1993
		DE 69229203 D	24-06-1999
		DE 69229203 T	09-12-1999
		JP 2119125 C	06-12-1996
		JP 5210570 A	20-08-1993
		JP 8031060 B	27-03-1996
		MX 9205088 A	01-03-1993
US 5566308 A	15-10-1996	US 5915266 A	22-06-1999
US 4679140 A	07-07-1987	JP 61084735 A	30-04-1986
		JP 60134937 A	18-07-1985
		DE 3479356 D	14-09-1989
		EP 0148478 A	17-07-1985